Jenkins Continuous Integration Tools

Pros:
-Free, open source CI tool
-Allows automation of build and tests.
-Local app
-Lots of features
-Well-established and good reputation.
-Handles multiple different types of security authentications

Jenkins being completely free makes it an excellent choice for a smaller/startup company that maybe cannot afford to dive into some of the more expensive tools. The documentation for jenkins seems to be relatively lightweight and easy to read, which leads me to believe that the setup and usage should go relatively smoothly. There are a number of jenkins-specific tools with their own full chapters of documentation. Jenkins has been around since early 2011 as a CI platform that can also be configured for CD, and the most recent stable release was on 25 Aug 2021 so it seems to be fairly continuously updated/upgraded as well.

Sentry Error Monitoring

Pros:
-Documentation for 18 languages
-One of the oldest app monitoring tools
-Popular(easier to find and work with people who also use the app)
-Associates errors with specific releases
-Breadcrumbs shows you all events leading to errors
-Massive traceability
-Secure
-Visualization of errors

Sentry is, by all accounts, a very robust error monitoring tool. It can be tried for free but is paid after some time, but the paid features are well worth it to anyone interested in keeping their applications running error free. Sentry's documentation is far-reaching, covering 18 different languages which is likely to make integration smooth and easy for any company wishing to use it. Step-by-step setup instructions are very precisely worded as well. Sentry appears to be a growing/up and coming company with a solid team of employees stretched around the world.

ArraySize	doublerAppend() time	doublerInsert() time
tiny	76.708 µs	32.208 µs
small	89.875 µs	41.208 µs
medium	127.417 µs	184.625 µs
large	659.417 µs	10.142209 ms
extra large	4.068ms	1.0753 s

These results indicate that appending scales much better, in spite of the fact that the unshift function starts off being a quicker method. My assumptions as to why this is the case are that as unshift continues to get more and more additions to the array, it must continuously move everything within the array. At the low numbers this has an insignificant impact on performance, but when dealing with progressively more and more massive numbers this leads to progressively more and more numbers needing to be shifted. To add an item to the beginning of a 1 million item array, you must shift all of the million existing items, whereas the appending method simply finds the length of the array, and adds one to it while assigning that index a new value.